

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

As demonstrated in the chart below, ASUS directly and indirectly infringes at least claim 9 of US 10,536,714 (the “‘714 Patent”). ASUS directly infringes, contributes to the infringement of, and/or induces infringement of the ‘714 Patent by making, using, selling, offering for sale, and/or importing into the United States the Accused Products that are covered by one or more claims of the ‘714 Patent. The Accused Products are devices that decode H.265-compliant video and/or encode video into H.265-compliant formats. For example, the ASUS Q543MV Notebook (“ASUS Q543MV”) is a representative product for other ASUS devices that decode H.265-compliant video and/or encode video into H.265-compliant formats.

The ASUS Q543MV contains at least one video decoder that helps decode H.265-compliant video. Additionally, the ASUS Q543MV contains at least one video encoder that helps encode video into H.265-compliant video formats.<sup>1</sup> While evidence from the ASUS Q543MV is specifically charted herein, the evidence and contentions charted herein apply equally to the other ASUS Accused Products that decode H.265-compliant video. On information and belief, the evidence and contentions charted herein apply equally to the other ASUS Accused Products that encode video into H.265-compliant formats.

No part of this exemplary chart construes, or is intended to construe, the specification, file history, or claims of the ‘714 Patent. Moreover, this exemplary chart does not limit, and is not intended to limit, Nokia’s infringement positions or contentions.

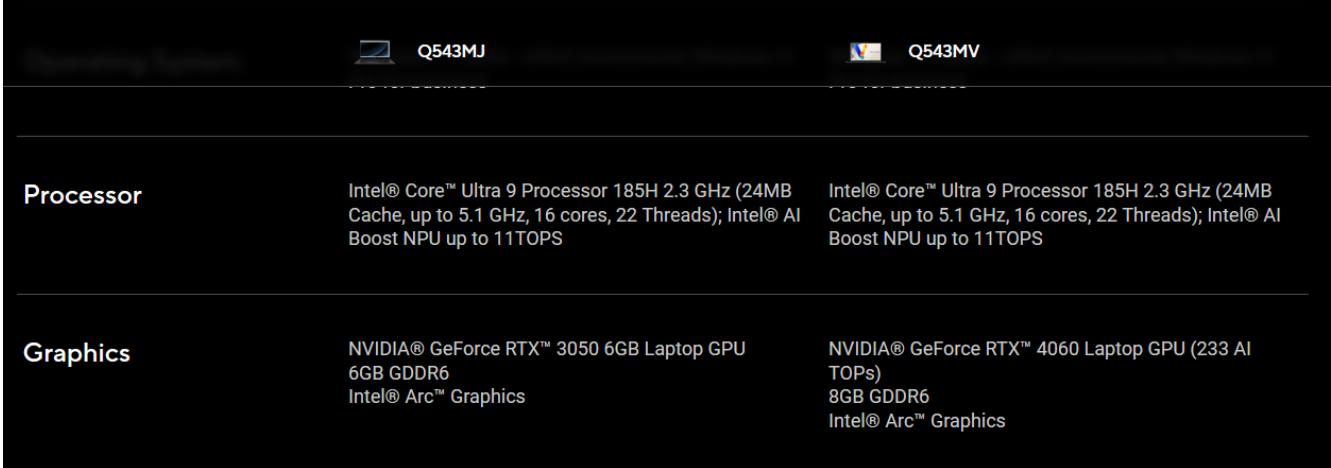
The following infringement chart includes exemplary citations to ITU-T Rec. H.265 (12/2016) High efficiency video coding (available at <https://www.itu.int/rec/T-REC-H.265-201612-S/en>) (the “H.265 Standard”). The cited functionality has been included in editions of the H.265 Standard since April 2013 and remains in current editions of the H.265 Standard. Any ASUS device that includes a decoder that practices the functionality in any of these editions of the H.265 Standard (“H.265 Decoder”) practices claims of the ‘714 Patent. Thus, the Accused Products each practice the H.265 Standard and are covered by claims of the ‘714 Patent.

Nokia contends each of the following limitations is met literally, and, to the extent a limitation is not met literally, it is met under the doctrine of equivalents.<sup>2</sup>

<sup>1</sup> See, e.g., <https://www.asus.com/us/laptops/for-home/everyday-use/asus-vivobook-pro-15-oled-q543/techspec/>; <https://www.intel.com/content/www/us/en/products/sku/236849/intel-core-ultra-9-processor-185h-24m-cache-up-to-5-10-ghz/specifications.html>; <https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new>.

<sup>2</sup> This claim chart is based on the information currently available to Nokia and is intended to be exemplary in nature. Nokia reserves all rights to update and elaborate its infringement positions, including as Nokia obtains additional information during discovery.

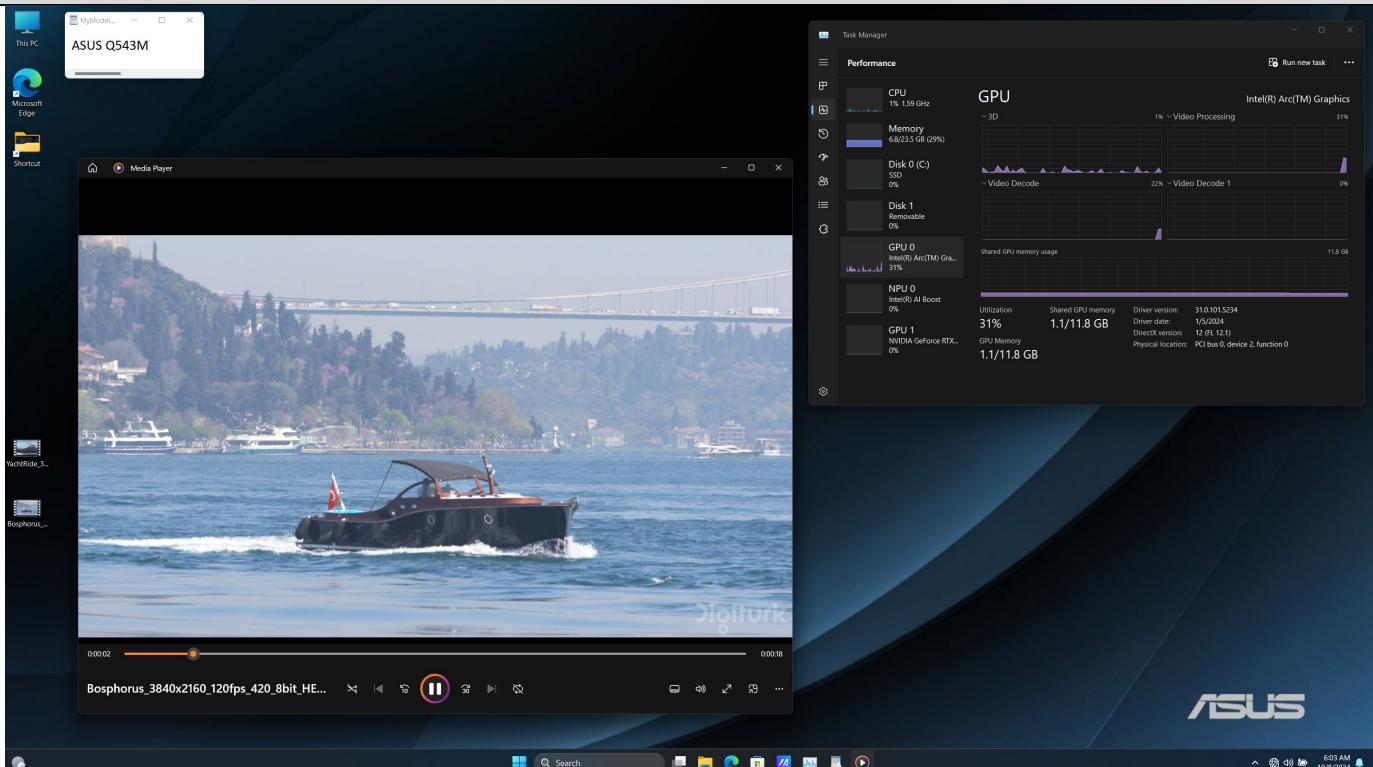
**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

<b>U.S. PATENT No. 10,536,714</b>	<b>ASUS ACCUSED PRODUCTS</b>							
<b>9. [A] A method comprising:</b>	<p>Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising the limitations below.</p> <p>For example, and without limitation, the Asus Q543MV uses hardware-accelerate video decoding and includes an NVIDIA GeForce RTX 4060 Laptop graphics processing unit (“GPU”) and an Intel Core Ultra 9 Processor 185H.</p>  <table border="1" style="margin-top: 10px;"> <tr> <td><b>Processor</b></td> <td>Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS</td> <td>Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS</td> </tr> <tr> <td><b>Graphics</b></td> <td>NVIDIA® GeForce RTX™ 3050 6GB Laptop GPU 6GB GDDR6 Intel® Arc™ Graphics</td> <td>NVIDIA® GeForce RTX™ 4060 Laptop GPU (233 AI TOPS) 8GB GDDR6 Intel® Arc™ Graphics</td> </tr> </table> <p>Source: <a href="https://www.asus.com/us/laptops/for-home/everyday-use/asus-vivobook-pro-15-oled-q543/techspec/">https://www.asus.com/us/laptops/for-home/everyday-use/asus-vivobook-pro-15-oled-q543/techspec/</a> (last accessed March 6, 2025).</p>		<b>Processor</b>	Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS	Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS	<b>Graphics</b>	NVIDIA® GeForce RTX™ 3050 6GB Laptop GPU 6GB GDDR6 Intel® Arc™ Graphics	NVIDIA® GeForce RTX™ 4060 Laptop GPU (233 AI TOPS) 8GB GDDR6 Intel® Arc™ Graphics
<b>Processor</b>	Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS	Intel® Core™ Ultra 9 Processor 185H 2.3 GHz (24MB Cache, up to 5.1 GHz, 16 cores, 22 Threads); Intel® AI Boost NPU up to 11TOPS						
<b>Graphics</b>	NVIDIA® GeForce RTX™ 3050 6GB Laptop GPU 6GB GDDR6 Intel® Arc™ Graphics	NVIDIA® GeForce RTX™ 4060 Laptop GPU (233 AI TOPS) 8GB GDDR6 Intel® Arc™ Graphics						
	H.264 Hardware Encode/Decode <span style="color: blue;">?</span>	Yes						
	H.265 (HEVC) Hardware Encode/Decode <span style="color: blue;">?</span>	Yes						
	AV1 Encode/Decode <span style="color: blue;">?</span>	Yes						

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714		ASUS ACCUSED PRODUCTS																																																																		
		Source: <a href="https://www.intel.com/content/www/us/en/products/sku/236849/intel-core-ultra-9-processor-185h-24m-cache-up-to-5-10-ghz/specifications.html">https://www.intel.com/content/www/us/en/products/sku/236849/intel-core-ultra-9-processor-185h-24m-cache-up-to-5-10-ghz/specifications.html</a> (last accessed March 6, 2025)(specifications for Intel Core Ultra 9 185H).																																																																		
<table border="1"> <thead> <tr> <th># BOARD</th><th># FAMILY</th><th># NVENC Generation</th><th>Desktop/ Mobile</th><th># OF CHIPS</th><th>Total # of NVENC</th><th>Max # of concurrent sessions</th><th>H.264 (AVCHD) YUV 4:2:0</th><th>H.264 (AVCHD) YUV 4:2:2</th><th>H.264 (AVCHD) YUV 4:4:4</th><th>H.264 (AVCHD) Lossless</th><th>H.265 (HEVC) 4K YUV 4:2:0</th><th>H.265 (HEVC) 4:2:2</th><th>H.265 (HEVC)</th><th>H.265 (HEVC) 4K YUV .</th></tr> </thead> <tbody> <tr> <td>GeForce RTX 4060 Laptop</td><td>Ada Lovelace</td><td>8th Gen</td><td>M</td><td>1</td><td>1</td><td>8</td><td>YES</td><td>NO</td><td>YES</td><td>YES</td><td>YES</td><td>NO</td><td>YES</td><td></td><td></td><td></td><td></td></tr> <tr> <td>GeForce RTX 4060</td><td>Ada Lovelace</td><td>8th Gen</td><td>D</td><td>1</td><td>1</td><td>8</td><td>YES</td><td>NO</td><td>YES</td><td>YES</td><td>YES</td><td>NO</td><td>YES</td><td></td><td></td><td></td><td></td></tr> </tbody> </table>																		# BOARD	# FAMILY	# NVENC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVENC	Max # of concurrent sessions	H.264 (AVCHD) YUV 4:2:0	H.264 (AVCHD) YUV 4:2:2	H.264 (AVCHD) YUV 4:4:4	H.264 (AVCHD) Lossless	H.265 (HEVC) 4K YUV 4:2:0	H.265 (HEVC) 4:2:2	H.265 (HEVC)	H.265 (HEVC) 4K YUV .	GeForce RTX 4060 Laptop	Ada Lovelace	8th Gen	M	1	1	8	YES	NO	YES	YES	YES	NO	YES					GeForce RTX 4060	Ada Lovelace	8th Gen	D	1	1	8	YES	NO	YES	YES	YES	NO	YES				
# BOARD	# FAMILY	# NVENC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVENC	Max # of concurrent sessions	H.264 (AVCHD) YUV 4:2:0	H.264 (AVCHD) YUV 4:2:2	H.264 (AVCHD) YUV 4:4:4	H.264 (AVCHD) Lossless	H.265 (HEVC) 4K YUV 4:2:0	H.265 (HEVC) 4:2:2	H.265 (HEVC)	H.265 (HEVC) 4K YUV .																																																						
GeForce RTX 4060 Laptop	Ada Lovelace	8th Gen	M	1	1	8	YES	NO	YES	YES	YES	NO	YES																																																							
GeForce RTX 4060	Ada Lovelace	8th Gen	D	1	1	8	YES	NO	YES	YES	YES	NO	YES																																																							
<table border="1"> <thead> <tr> <th># BOARD</th><th># FAMILY</th><th># NVDEC Generation</th><th>Desktop/ Mobile</th><th># OF CHIPS</th><th>Total # of NVDEC</th><th>MPEG-1</th><th>MPEG-2</th><th>VC-1</th><th>VP8</th><th>VP9 4:2:0</th><th>H.264 (AVCHD) 4:2:0</th><th>8 Bit</th><th>10 Bit</th><th>12 Bit</th><th>8 Bit</th><th>10 Bit</th></tr> </thead> <tbody> <tr> <td>GeForce RTX 4060 Laptop</td><td>Ada Lovelace</td><td>5th Gen</td><td>M</td><td>1</td><td>1</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>NO</td><td></td></tr> </tbody> </table>																		# BOARD	# FAMILY	# NVDEC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVDEC	MPEG-1	MPEG-2	VC-1	VP8	VP9 4:2:0	H.264 (AVCHD) 4:2:0	8 Bit	10 Bit	12 Bit	8 Bit	10 Bit	GeForce RTX 4060 Laptop	Ada Lovelace	5th Gen	M	1	1	YES	NO																										
# BOARD	# FAMILY	# NVDEC Generation	Desktop/ Mobile	# OF CHIPS	Total # of NVDEC	MPEG-1	MPEG-2	VC-1	VP8	VP9 4:2:0	H.264 (AVCHD) 4:2:0	8 Bit	10 Bit	12 Bit	8 Bit	10 Bit																																																				
GeForce RTX 4060 Laptop	Ada Lovelace	5th Gen	M	1	1	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	NO																																																				
<table border="1"> <thead> <tr> <th colspan="3">H.265 (HEVC) 4:2:0</th><th colspan="3">H.265 (HEVC) 4:2:2</th><th colspan="3">H.265 (HEVC) 4:4:4</th><th colspan="3">AV1</th></tr> <tr> <th>8 Bit</th><th>10 Bit</th><th>12 Bit</th><th>8 Bit</th><th>10 Bit</th><th>8 Bit</th><th>10 Bit</th><th>12 Bit</th><th>8 Bit</th><th>10 Bit</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></tr> </thead> <tbody> <tr> <td>YES</td><td>YES</td><td>YES</td><td>NO</td><td>NO</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td>YES</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr> </tbody> </table>																		H.265 (HEVC) 4:2:0			H.265 (HEVC) 4:2:2			H.265 (HEVC) 4:4:4			AV1			8 Bit	10 Bit	12 Bit	8 Bit	10 Bit	8 Bit	10 Bit	12 Bit	8 Bit	10 Bit									YES	YES	YES	NO	NO	YES	YES	YES	YES	YES											
H.265 (HEVC) 4:2:0			H.265 (HEVC) 4:2:2			H.265 (HEVC) 4:4:4			AV1																																																											
8 Bit	10 Bit	12 Bit	8 Bit	10 Bit	8 Bit	10 Bit	12 Bit	8 Bit	10 Bit																																																											
YES	YES	YES	NO	NO	YES	YES	YES	YES	YES																																																											
Source: <a href="https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new">https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new</a> (last accessed March 6, 2025)(row for 4060 Laptop GPU).																																																																				
For example, an ASUS Q543MV was used to playback an H.265-compliant video.																																																																				

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	 <p>Source: Screenshot of H.265-compliant video playback on ASUS Q543MV.</p> <p>For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. Each of the ASUS Accused Products performs a method comprising the limitations below.</p> <p><b>3 Definitions</b></p> <p>For the purposes of this Recommendation   International Standard, the following definitions apply.</p> <p>...</p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p><b>3.12 bitstream:</b> A sequence of bits, . . . , that forms the representation of <i>coded pictures</i> and associated data forming one or more coded video sequences (<i>CVSs</i>).</p> <p>...</p> <p><b>3.25 coded picture:</b> A <i>coded representation of a picture</i> . . .</p> <p>...</p> <p><b>3.44 decoding process:</b> The process specified in this Specification that reads a <i>bitstream</i> and derives <i>decoded pictures</i> from it.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 4 – 7.</p>
<p>[B] selecting a first spatial motion vector prediction candidate from a set of spatial motion vector prediction candidates for an encoded block of pixels as a potential spatial motion vector prediction candidate to be included in a motion vector prediction list for a prediction unit of the encoded block of pixels, where the motion vector prediction list comprises motion information of the spatial motion vector prediction candidates;</p>	<p>Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising selecting a first spatial motion vector prediction candidate from a set of spatial motion vector prediction candidates for an encoded block of pixels as a potential spatial motion vector prediction candidate to be included in a motion vector prediction list for a prediction unit of the encoded block of pixels, where the motion vector prediction list comprises motion information of the spatial motion vector prediction candidates.</p> <p>For example, and without limitation, the H.265 Standard specifies the following regarding the decoding process. The Accused Products perform a method comprising selecting a first spatial motion vector prediction candidate from a set of spatial motion vector prediction candidates for an encoded block of pixels as a potential spatial motion vector prediction candidate to be included in a motion vector prediction list for a prediction unit of the encoded block of pixels, where the motion vector prediction list comprises motion information of the spatial motion vector prediction candidates, corresponding to the decoding process specified by the H.265 Standard.</p> <p>As specified in Subclause 8.5.3.2.3 of the H.265 Standard, the spatial motion vector prediction candidates A0, A1, B0, B1, and B2 (see Figure 8-3) are processed in the order A1, B1, B0, A0, and B2. For each location A1, B1, B0, A0, and B2, the Accused Products check the availability of the block as specified in Subclause 6.4.2. If a block is coded in intra-coding or not available (e.g., the block is outside of the current</p>

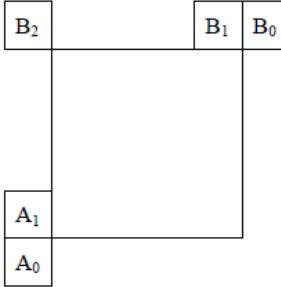
**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

<b>U.S. PATENT No. 10,536,714</b>	<b>ASUS ACCUSED PRODUCTS</b>
	<p>slice or tile), its availability flag is set to FALSE, and it is not considered as a candidate for motion vector prediction and it is not added to the candidate list.</p> <p>The following specifications provide further evidence of how each of the Accused Products operates:</p> <p style="text-align: center;"><b>8.5.3.2.3 Derivation process for spatial merging candidates</b></p> <p style="text-align: center;">...</p> <p>For the derivation of availableFlagA<sub>1</sub>, refIdxLXA<sub>1</sub>, predFlagLXA<sub>1</sub> and mvLXA<sub>1</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) inside the neighbouring luma coding block is set equal to ( xPb – 1, yPb + nPbH ).</li> <li>– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location ( xCb, yCb ), the current luma coding block size nCbS, the luma prediction block location ( xPb, yPb ), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag availableA<sub>1</sub>.</li> </ul> <p style="text-align: center;">...</p> <p>For the derivation of availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW – 1, yPb – 1 ).</li> <li>– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location ( xCb, yCb ), the current luma coding block size nCbS, the luma prediction block location ( xPb, yPb ), the luma prediction block width nPbW, the luma prediction</li> </ul>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>block height nPbH, the luma location ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag availableB<sub>1</sub>.</p> <p>...</p> <p>For the derivation of availableFlagB<sub>0</sub>, refIdxLXB<sub>0</sub>, predFlagLXB<sub>0</sub> and mvLXB<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW, yPb – 1 ).</li> <li>– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location ( xCb, yCb ), the current luma coding block size nCbS, the luma prediction block location ( xPb, yPb ), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag availableB<sub>0</sub>.</li> </ul> <p>...</p> <p>For the derivation of availableFlagA<sub>0</sub>, refIdxLXA<sub>0</sub>, predFlagLXA<sub>0</sub> and mvLXA<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbA<sub>0</sub>, yNbA<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb – 1, yPb + nPbH ).</li> <li>– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location ( xCb, yCb ), the current luma coding block size nCbS, the luma prediction block location ( xPb, yPb ), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ( xNbA<sub>0</sub>, yNbA<sub>0</sub> ) and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag availableA<sub>0</sub>.</li> </ul> <p>...</p> <p>For the derivation of availableFlagB<sub>2</sub>, refIdxLXB<sub>2</sub>, predFlagLXB<sub>2</sub> and mvLXB<sub>2</sub> the following applies:</p>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<ul style="list-style-type: none"> <li>– The luma location ( <math>xNbB2, yNbB2</math> ) inside the neighbouring luma coding block is set equal to ( <math>xPb - 1, yPb - 1</math> ).</li> <li>– The availability derivation process for a prediction block as specified in clause 6.4.2 is invoked with the luma location ( <math>xCb, yCb</math> ), the current luma coding block size <math>nCbS</math>, the luma prediction block location ( <math>xPb, yPb</math> ), the luma prediction block width <math>nPbW</math>, the luma prediction block height <math>nPbH</math>, the luma location ( <math>xNbB2, yNbB2</math> ) and the partition index <math>partIdx</math> as inputs, and the output is assigned to the prediction block availability flag <math>availableB2</math>.</li> </ul> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.</p> <p><b>8.5.3.2.7 Derivation process for motion vector predictor candidates</b></p> <p>...</p>  <p>Figure 8-3 – Spatial motion vector neighbours (informative)</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 156.</p>
[C] determining a subset of spatial motion vector prediction candidates based on the location of the block	Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising determining a subset of spatial motion vector prediction candidates based on the location of the block associated with the first spatial motion vector prediction candidate.

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

<b>U.S. PATENT NO. 10,536,714</b>	<b>ASUS ACCUSED PRODUCTS</b>
associated with the first spatial motion vector prediction candidate;	<p>For example, the spatial motion vector prediction candidates A1, B1, B0, A0, and B2 are processed in this order. For example, when the Accused Products select spatial motion vector prediction candidate at position B2 as a potential candidate, they determine a subset of candidates as (B1, A1). <i>See Steps 8 and 9 in Subclause 8.5.3.2.3.</i> In another example, when B0 is selected, the Accused Product determine the subset as (B1). <i>See Step 4 in Subclause 8.5.3.2.3.</i> In yet another example, when A0 is selected, the Accused Products determine the subset as (A1). <i>See Step 6 in Subclause 8.5.3.2.3; see Subclause 8.5.3.2.3.</i></p> <p><b>8.5.3.2.3 Derivation process for spatial merging candidates</b></p> <p>...</p> <p>For the derivation of availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW – 1, yPb – 1 ).</li> <li>...</li> <li>– The variables availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> are derived as follows: <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlagB<sub>1</sub> is set equal to 0, both components of mvLXB<sub>1</sub> are set equal to 0, refIdxLXB<sub>1</sub> is set equal to –1 and predFlagLXB<sub>1</sub> is set equal to 0, with X being 0 or 1: <ul style="list-style-type: none"> <li>...</li> <li>2. availableA<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) and ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) have the same motion vectors and the same reference indices.</li> <li>...</li> </ul> </li> </ul> </li> </ul> <p>For the derivation of availableFlagB<sub>0</sub>, refIdxLXB<sub>0</sub>, predFlagLXB<sub>0</sub> and mvLXB<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW, yPb – 1 ).</li> </ul>

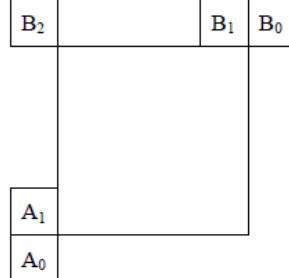
**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>...</p> <ul style="list-style-type: none"> <li>– The variables availableFlagB<sub>0</sub>, refIdxLXB<sub>0</sub>, predFlagLXB<sub>0</sub> and mvLXB<sub>0</sub> are derived as follows:           <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlagB<sub>0</sub> is set equal to 0, both components of mvLXB<sub>0</sub> are set equal to 0, refIdxLXB<sub>0</sub> is set equal to -1 and predFlagLXB<sub>0</sub> is set equal to 0, with X being 0 or 1:               <p>...</p> <p>4. availableB<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) and ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) have the same motion vectors and the same reference indices.</p> <p>...</p> </li> </ul> </li> </ul> <p>For the derivation of availableFlagA<sub>0</sub>, refIdxLXA<sub>0</sub>, predFlagLXA<sub>0</sub> and mvLXA<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbA<sub>0</sub>, yNbA<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb - 1, yPb + nPbH ).</li> </ul> <p>...</p> <ul style="list-style-type: none"> <li>– The variables availableFlagA<sub>0</sub>, refIdxLXA<sub>0</sub>, predFlagLXA<sub>0</sub> and mvLXA<sub>0</sub> are derived as follows:           <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlagA<sub>0</sub> is set equal to 0, both components of mvLXA<sub>0</sub> are set equal to 0, refIdxLXA<sub>0</sub> is set equal to -1 and predFlagLXA<sub>0</sub> is set equal to 0, with X being 0 or 1:               <p>...</p> <p>6. availableA<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) and ( xNbA<sub>0</sub>, yNbA<sub>0</sub> ) have the same motion vectors and the same reference indices.</p> <p>...</p> </li> </ul> </li> </ul> <p>For the derivation of availableFlagB<sub>2</sub>, refIdxLXB<sub>2</sub>, predFlagLXB<sub>2</sub> and mvLXB<sub>2</sub> the following applies:</p>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<ul style="list-style-type: none"> <li>– The luma location ( <math>xNbB_2</math>, <math>yNbB_2</math> ) inside the neighbouring luma coding block is set equal to ( <math>xPb - 1</math>, <math>yPb - 1</math> ).</li> <li>...</li> <li>– The variables availableFlag<math>B_2</math>, refIdxLXB<math>_2</math>, predFlagLXB<math>_2</math> and mvLXB<math>_2</math> are derived as follows:           <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlag<math>B_2</math> is set equal to 0, both components of mvLXB<math>_2</math> are set equal to 0, refIdxLXB<math>_2</math> is set equal to -1 and predFlagLXB<math>_2</math> is set equal to 0, with X being 0 or 1:</li> <li>...</li> <li>8. availableA<math>_1</math> is equal to TRUE and prediction units covering the luma locations ( <math>xNbA_1</math>, <math>yNbA_1</math> ) and ( <math>xNbB_2</math>, <math>yNbB_2</math> ) have the same motion vectors and the same reference indices.</li> <li>9. availableB<math>_1</math> is equal to TRUE and the prediction units covering the luma locations ( <math>xNbB_1</math>, <math>yNbB_1</math> ) and ( <math>xNbB_2</math>, <math>yNbB_2</math> ) have the same motion vectors and the same reference indices.</li> <li>...</li> <li>...</li> </ul> </li> </ul> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.</p> <p><b>8.5.3.2.7 Derivation process for motion vector predictor candidates</b></p> <p>...</p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	 <p style="text-align: center;"><b>Figure 8-3 – Spatial motion vector neighbours (informative)</b></p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156</p>
<p>[D] comparing motion information of the first spatial motion vector prediction candidate with motion information of another spatial motion vector prediction candidate of the set of spatial motion vector prediction candidates without making a comparison of each possible candidate pair from the set of spatial motion vector prediction candidates;</p>	<p>Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising comparing motion information of the first spatial motion vector prediction candidate with motion information of another spatial motion vector prediction candidate of the set of spatial motion vector prediction candidates without making a comparison of each possible candidate pair from the set of spatial motion vector prediction candidates.</p> <p>For example, when considering the spatial motion vector prediction candidate at position B2, the Accused Products compare motion information for spatial motion vector prediction candidate at position B2 with motion information of spatial motion vector prediction candidates in the subset (B1, A1) of spatial motion vector prediction candidates. <i>See Steps 8 and 9 in Subclause 8.5.3.2.3.</i> Motion information of spatial motion vector prediction candidate at position B2 is not compared with motion information of A0, and B0. In this example, the Accused Products checks whether motion information at position B2 is equal to motion information at position B1 and motion information at position A1. <i>See Steps 8 and 9 in Subclause 8.5.3.2.3.</i> If motion information at position B2 is equal to motion information at either position B1 or A1, then B2 will not be included in the list.</p> <p>As another example, the Accused Products compare motion information for spatial motion vector prediction candidate at position A0 with motion information of spatial motion vector prediction candidates in the subset</p>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>(A1) of spatial motion vector prediction candidates. <i>See Step 6 in Subclause 8.5.3.2.3.</i> Motion information of spatial motion vector prediction candidate at position A0 is not compared with motion information of B1, B0, and B2. In this example, the Accused Products check whether motion information at position A0 is equal to motion information at position A1. <i>See Step 6 in Subclause 8.5.3.2.3.</i> If motion information at position A0 is equal to motion information at position A1, then A0 will not be included in the list.</p> <p><b>8.5.3.2.3 Derivation process for spatial merging candidates</b></p> <p>...</p> <p>For the derivation of availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW – 1, yPb – 1 ).</li> <li>...</li> <li>– The variables availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> are derived as follows:</li> <li>– If one or more of the following conditions are true, availableFlagB<sub>1</sub> is set equal to 0, both components of mvLXB<sub>1</sub> are set equal to 0, refIdxLXB<sub>1</sub> is set equal to –1 and predFlagLXB<sub>1</sub> is set equal to 0, with X being 0 or 1:</li> <li>...</li> <li>1. availableB<sub>1</sub> is equal to FALSE.</li> <li>2. availableA<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) and ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) have the same motion vectors and the same reference indices.</li> <li>– Otherwise, availableFlagB<sub>1</sub> is set equal to 1 and the following assignments are made:</li> </ul> $\text{mvLXB1} = \text{MvLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-131)$ $\text{refIdxLXB1} = \text{RefIdxLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-132)$

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p><math>\text{predFlagLXB1} = \text{PredFlagLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-133)</math></p> <p>For the derivation of <math>\text{availableFlagB}_0</math>, <math>\text{refIdxLXB}_0</math>, <math>\text{predFlagLXB}_0</math> and <math>\text{mvLXB}_0</math> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location (<math>\text{xNbB}_0</math>, <math>\text{yNbB}_0</math>) inside the neighbouring luma coding block is set equal to (<math>\text{xPb} + \text{nPbW}</math>, <math>\text{yPb} - 1</math>).</li> <li>...</li> <li>– The variables <math>\text{availableFlagB}_0</math>, <math>\text{refIdxLXB}_0</math>, <math>\text{predFlagLXB}_0</math> and <math>\text{mvLXB}_0</math> are derived as follows: <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, <math>\text{availableFlagB}_0</math> is set equal to 0, both components of <math>\text{mvLXB}_0</math> are set equal to 0, <math>\text{refIdxLXB}_0</math> is set equal to -1 and <math>\text{predFlagLXB}_0</math> is set equal to 0, with X being 0 or 1: <ul style="list-style-type: none"> <li>...</li> <li>3. <math>\text{availableB}_0</math> is equal to FALSE.</li> <li>4. <math>\text{availableB}_1</math> is equal to TRUE and the prediction units covering the luma locations (<math>\text{xNbB}_1</math>, <math>\text{yNbB}_1</math>) and (<math>\text{xNbB}_0</math>, <math>\text{yNbB}_0</math>) have the same motion vectors and the same reference indices.</li> <li>– Otherwise, <math>\text{availableFlagB}_0</math> is set equal to 1 and the following assignments are made:</li> </ul> </li> </ul> </li> </ul> <p><math>\text{mvLXB0} = \text{MvLX}[ \text{xNbB0} ][ \text{yNbB0} ] \quad (8-134)</math></p> <p><math>\text{refIdxLXB0} = \text{RefIdxLX}[ \text{xNbB0} ][ \text{yNbB0} ] \quad (8-135)</math></p> <p><math>\text{predFlagLXB0} = \text{PredFlagLX}[ \text{xNbB0} ][ \text{yNbB0} ] \quad (8-136)</math></p> <p>For the derivation of <math>\text{availableFlagA}_0</math>, <math>\text{refIdxLXA}_0</math>, <math>\text{predFlagLXA}_0</math> and <math>\text{mvLXA}_0</math> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location (<math>\text{xNbA}_0</math>, <math>\text{yNbA}_0</math>) inside the neighbouring luma coding block is set equal to (<math>\text{xPb} - 1</math>, <math>\text{yPb} + \text{nPbH}</math>).</li> </ul>

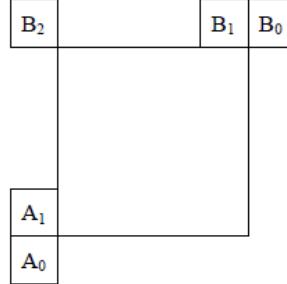
**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>...</p> <ul style="list-style-type: none"> <li>– The variables availableFlagA0, refIdxLXA0, predFlagLXA0 and mvLXA0 are derived as follows:</li> <li>– If one or more of the following conditions are true, availableFlagA0 is set equal to 0, both components of mvLXA0 are set equal to 0, refIdxLXA0 is set equal to -1 and predFlagLXA0 is set equal to 0, with X being 0 or 1:</li> </ul> <p>5. availableA0 is equal to FALSE.</p> <p>6. availableA1 is equal to TRUE and the prediction units covering the luma locations ( xNbA1, yNbA1 ) and ( xNbA0, yNbA0 ) have the same motion vectors and the same reference indices.</p> <ul style="list-style-type: none"> <li>– Otherwise, availableFlagA0 is set equal to 1 and the following assignments are made:</li> </ul> $\text{mvLXA0} = \text{MvLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-137)$ $\text{refIdxLXA0} = \text{RefIdxLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-138)$ $\text{predFlagLXA0} = \text{PredFlagLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-139)$ <p>For the derivation of availableFlagB2, refIdxLXB2, predFlagLXB2 and mvLXB2 the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB2, yNbB2 ) inside the neighbouring luma coding block is set equal to ( xPb - 1, yPb - 1 ).</li> <li>...</li> <li>– The variables availableFlagB2, refIdxLXB2, predFlagLXB2 and mvLXB2 are derived as follows:</li> </ul>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>– If one or more of the following conditions are true, availableFlagB2 is set equal to 0, both components of mvLXB2 are set equal to 0, refIdxLXB2 is set equal to -1 and predFlagLXB2 is set equal to 0, with X being 0 or 1:</p> <p>7. availableB2 is equal to FALSE.</p> <p>8. availableA1 is equal to TRUE and prediction units covering the luma locations ( xNbA1, yNbA1 ) and ( xNbB2, yNbB2 ) have the same motion vectors and the same reference indices.</p> <p>9. availableB1 is equal to TRUE and the prediction units covering the luma locations ( xNbB1, yNbB1 ) and ( xNbB2, yNbB2 ) have the same motion vectors and the same reference indices.</p> <p>10. availableFlagA0 + availableFlagA1 + availableFlagB0 + availableFlagB1 is equal to 4.</p> <p>– Otherwise, availableFlagB2 is set equal to 1 and the following assignments are made:</p> <p style="padding-left: 40px;">mvLXB2 = MvLX[ xNbB2 ][ yNbB2 ] (8-140)</p> <p style="padding-left: 40px;">refIdxLXB2 = RefIdxLX[ xNbB2 ][ yNbB2 ] (8-141)</p> <p style="padding-left: 40px;">predFlagLXB2 = PredFlagLX[ xNbB2 ][ yNbB2 ] (8-142)</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.</p> <p><b>8.5.3.2.7 Derivation process for motion vector predictor candidates</b></p> <p style="text-align: center;">...</p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	 <p style="text-align: center;"><b>Figure 8-3 – Spatial motion vector neighbours (informative)</b></p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156.</p>
<p>[E] determining to include or exclude the first spatial motion vector prediction candidate in the motion vector prediction list based on the comparing; and</p>	<p>Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising determining to include or exclude the first spatial motion vector prediction candidate in the motion vector prediction list based on the comparing.</p> <p>For example, and as explained above, when considering the spatial motion vector prediction candidate at position B2, the Accused Products compare motion information for spatial motion vector prediction candidate at position B2 with motion information of spatial motion vector prediction candidates in the subset (B1, A1) of spatial motion vector prediction candidates. <i>See Steps 8 and 9 in Subclause 8.5.3.2.3.</i> Motion information of spatial motion vector prediction candidate at position B2 is not compared with motion information of A0, and B0. In this example, the Accused Products checks whether motion information at position B2 is equal to motion information at position B1 and motion information at position A1. <i>See Steps 8 and 9 in Subclause 8.5.3.2.3.</i> If motion information at position B2 is equal to motion information at either position B1 or A1, then B2 will not be included in the list.</p> <p>As another example, and as also explained above, the Accused Products compare motion information for spatial motion vector prediction candidate at position A0 with motion information of spatial motion vector prediction candidates in the subset (A1) of spatial motion vector prediction candidates. <i>See Step 6 in Subclause 8.5.3.2.3.</i> Motion information of spatial motion vector prediction candidate at position A0 is not</p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>compared with motion information of B1, B2, and B0. In this example, the Accused Products check whether motion information at position A0 is equal to motion information at position A1. <i>See Step 6 in Subclause 8.5.3.2.3.</i> If motion information at position A0 is equal to motion information at position A1, then A0 will not be included in the list.</p> <p><b>8.5.3.2.3 Derivation process for spatial merging candidates</b></p> <p>...</p> <p>For the derivation of availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW – 1, yPb – 1 ).</li> <li>...</li> <li>– The variables availableFlagB<sub>1</sub>, refIdxLXB<sub>1</sub>, predFlagLXB<sub>1</sub> and mvLXB<sub>1</sub> are derived as follows:</li> <li>– If one or more of the following conditions are true, availableFlagB<sub>1</sub> is set equal to 0, both components of mvLXB<sub>1</sub> are set equal to 0, refIdxLXB<sub>1</sub> is set equal to –1 and predFlagLXB<sub>1</sub> is set equal to 0, with X being 0 or 1:</li> <li>...</li> <li>5. availableB<sub>1</sub> is equal to FALSE.</li> <li>6. availableA<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbA<sub>1</sub>, yNbA<sub>1</sub> ) and ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) have the same motion vectors and the same reference indices.</li> <li>– Otherwise, availableFlagB<sub>1</sub> is set equal to 1 and the following assignments are made:</li> </ul> $\text{mvLXB1} = \text{MvLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-131)$ $\text{refIdxLXB1} = \text{RefIdxLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-132)$ $\text{predFlagLXB1} = \text{PredFlagLX}[ \text{xNbB1} ][ \text{yNbB1} ] \quad (8-133)$

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>For the derivation of availableFlagB<sub>0</sub>, refIdxLXB<sub>0</sub>, predFlagLXB<sub>0</sub> and mvLXB<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb + nPbW, yPb – 1 ).</li> <li>...</li> <li>– The variables availableFlagB<sub>0</sub>, refIdxLXB<sub>0</sub>, predFlagLXB<sub>0</sub> and mvLXB<sub>0</sub> are derived as follows:</li> <li>– If one or more of the following conditions are true, availableFlagB<sub>0</sub> is set equal to 0, both components of mvLXB<sub>0</sub> are set equal to 0, refIdxLXB<sub>0</sub> is set equal to –1 and predFlagLXB<sub>0</sub> is set equal to 0, with X being 0 or 1:</li> <li>...</li> <li>7. availableB<sub>0</sub> is equal to FALSE.</li> <li>8. availableB<sub>1</sub> is equal to TRUE and the prediction units covering the luma locations ( xNbB<sub>1</sub>, yNbB<sub>1</sub> ) and ( xNbB<sub>0</sub>, yNbB<sub>0</sub> ) have the same motion vectors and the same reference indices.</li> <li>– Otherwise, availableFlagB<sub>0</sub> is set equal to 1 and the following assignments are made:</li> <p style="padding-left: 20px;">mvLXB<sub>0</sub> = MvLX[ xNbB<sub>0</sub> ][ yNbB<sub>0</sub> ] (8-134)</p> <p style="padding-left: 20px;">refIdxLXB<sub>0</sub> = RefIdxLX[ xNbB<sub>0</sub> ][ yNbB<sub>0</sub> ] (8-135)</p> <p style="padding-left: 20px;">predFlagLXB<sub>0</sub> = PredFlagLX[ xNbB<sub>0</sub> ][ yNbB<sub>0</sub> ] (8-136)</p> <p>For the derivation of availableFlagA<sub>0</sub>, refIdxLXA<sub>0</sub>, predFlagLXA<sub>0</sub> and mvLXA<sub>0</sub> the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbA<sub>0</sub>, yNbA<sub>0</sub> ) inside the neighbouring luma coding block is set equal to ( xPb – 1, yPb + nPbH ).</li> <li>...</li> </ul> </ul>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<ul style="list-style-type: none"> <li>– The variables availableFlagA0, refIdxLXA0, predFlagLXA0 and mvLXA0 are derived as follows:           <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlagA0 is set equal to 0, both components of mvLXA0 are set equal to 0, refIdxLXA0 is set equal to -1 and predFlagLXA0 is set equal to 0, with X being 0 or 1:               <ol style="list-style-type: none"> <li>5. availableA0 is equal to FALSE.</li> <li>6. availableA1 is equal to TRUE and the prediction units covering the luma locations ( xNbA1, yNbA1 ) and ( xNbA0, yNbA0 ) have the same motion vectors and the same reference indices.</li> </ol> </li> <li>– Otherwise, availableFlagA0 is set equal to 1 and the following assignments are made:               <math display="block">\text{mvLXA0} = \text{MvLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-137)</math> <math display="block">\text{refIdxLXA0} = \text{RefIdxLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-138)</math> <math display="block">\text{predFlagLXA0} = \text{PredFlagLX}[ \text{xNbA0} ][ \text{yNbA0} ] \quad (8-139)</math> </li> </ul> </li> </ul> <p>For the derivation of availableFlagB2, refIdxLXB2, predFlagLXB2 and mvLXB2 the following applies:</p> <ul style="list-style-type: none"> <li>– The luma location ( xNbB2, yNbB2 ) inside the neighbouring luma coding block is set equal to ( xPb - 1, yPb - 1 ).</li> <li>...</li> <li>– The variables availableFlagB2, refIdxLXB2, predFlagLXB2 and mvLXB2 are derived as follows:           <ul style="list-style-type: none"> <li>– If one or more of the following conditions are true, availableFlagB2 is set equal to 0, both components of mvLXB2 are set equal to 0, refIdxLXB2 is set equal to -1 and predFlagLXB2 is set equal to 0, with X being 0 or 1:</li> </ul> </li> </ul>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>7. availableB2 is equal to FALSE.</p> <p>8. availableA1 is equal to TRUE and prediction units covering the luma locations ( <math>xNbA1, yNbA1</math> ) and ( <math>xNbB2, yNbB2</math> ) have the same motion vectors and the same reference indices.</p> <p>9. availableB1 is equal to TRUE and the prediction units covering the luma locations ( <math>xNbB1, yNbB1</math> ) and ( <math>xNbB2, yNbB2</math> ) have the same motion vectors and the same reference indices.</p> <p>10. availableFlagA0 + availableFlagA1 + availableFlagB0 + availableFlagB1 is equal to 4.</p> <p>– Otherwise, availableFlagB2 is set equal to 1 and the following assignments are made:</p> $mvLXB2 = MvLX[ xNbB2 ][ yNbB2 ] \quad (8-140)$ $refIdxLXB2 = RefIdxLX[ xNbB2 ][ yNbB2 ] \quad (8-141)$ $predFlagLXB2 = PredFlagLX[ xNbB2 ][ yNbB2 ] \quad (8-142)$ <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 150-152.</p> <p><b>8.5.3.2.7 Derivation process for motion vector predictor candidates</b></p> <p>...</p> <pre> graph LR     B2["B2"] --- B1["B1"]     B2 --- B0["B0"]     B1 --- B0     A1["A1"] --- A0["A0"]     style A1 fill:#ccc,stroke:#000     style A0 fill:#ccc,stroke:#000     style B2 fill:#fff,stroke:#000     style B1 fill:#fff,stroke:#000     style B0 fill:#fff,stroke:#000   </pre> <p><b>Figure 8-3 – Spatial motion vector neighbours (informative)</b></p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 156.</p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
<p>[F] selecting a spatial motion vector prediction candidate from the motion vector prediction list for use in decoding the encoded block of pixels, wherein the spatial motion vector prediction candidate is selected from the motion vector prediction list using information that was received identifying a respective spatial motion vector prediction candidate from the motion vector prediction list constructed by an encoder.</p>	<p>Each of the Accused Products, such as the ASUS Q543MV, performs a method comprising selecting a spatial motion vector prediction candidate from the motion vector prediction list for use in decoding the encoded block of pixels, wherein the spatial motion vector prediction candidate is selected from the motion vector prediction list using information that was received identifying a respective spatial motion vector prediction candidate from the motion vector prediction list constructed by an encoder.</p> <p>For example, the Accused Products receive syntax element <i>merge_idx</i> from the bitstream. See Subclause 7.3.8.6. The Accused Products select the motion vector prediction candidate at position <i>merge_idx</i> in the merging candidate list <i>mergeCandList</i>. See Step 9 in Subclause 8.5.3.2.2. The Accused Products then use that motion vector in decoding the current block.</p> <p><b>3 Definitions</b></p> <p>For the purposes of this Recommendation   International Standard, the following definitions apply.</p> <p>...</p> <p><b>3.153 syntax element:</b> An element of data represented in the <i>bitstream</i>.</p> <p><b>3.154 syntax structure:</b> Zero or more <i>syntax elements</i> present together in the <i>bitstream</i> in a specified order.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 5, 7-12.</p> <p><b>5.10 Variables, syntax elements and tables</b></p> <p>Syntax elements in the bitstream are represented in <b>bold</b> type. Each syntax element is described by its name (all lower case letters with underscore characters), and one descriptor for its method of coded representation. The decoding process behaves according to the value of the syntax element and to the values of previously decoded syntax elements. When a value of a syntax element is used in the syntax tables or the text, it appears in regular (i.e., not bold) type.</p>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 18.</p> <p><b>7 Syntax and semantics</b></p> <p><b>7.1 Method of specifying syntax in tabular form</b></p> <p>The syntax tables specify a superset of the syntax of all allowed bitstreams . . .</p> <p>...</p> <p>... When <b>syntax_element</b> appears, it specifies that a syntax element is parsed from the bitstream . . .</p> <p>...</p> <p><b>7.2 Specification of syntax functions and descriptors</b></p> <p>...</p> <p>The following descriptors specify the parsing process of each syntax element:</p> <ul style="list-style-type: none"> <li>– ae(v): context-adaptive arithmetic entropy-coded syntax element. The parsing process for this descriptor is specified in clause 9.3.</li> </ul> <p>...</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 30-31.</p> <p><b>7.3.8.6 Prediction unit syntax</b></p>

**EXHIBIT 9****UNITED STATES PATENT NO. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<pre> prediction_unit( x0, y0, nPbW, nPbH ) {     if( cu_skip_flag[ x0 ][ y0 ] ) {         if( MaxNumMergeCand &gt; 1 )             merge_idx[ x0 ][ y0 ]         } else { /* MODE_INTER */             merge_flag[ x0 ][ y0 ]             if( merge_flag[ x0 ][ y0 ] ) {                 if( MaxNumMergeCand &gt; 1 )                     merge_idx[ x0 ][ y0 ]             } else {                 if( slice_type == B )                     inter_pred_idc[ x0 ][ y0 ]                 if( inter_pred_idc[ x0 ][ y0 ] != PRED_L1 ) {                     if( num_ref_idx_10_active_minus1 &gt; 0 )                         ref_idx_10[ x0 ][ y0 ]                     mvd_coding( x0, y0, 0 )                     mvp_10_flag[ x0 ][ y0 ]                 }                 if( inter_pred_idc[ x0 ][ y0 ] != PRED_L0 ) {                     if( num_ref_idx_11_active_minus1 &gt; 0 )                         ref_idx_11[ x0 ][ y0 ]                     if( mvd_11_zero_flag &amp;&amp; inter_pred_idc[ x0 ][ y0 ] == PRED_BI ) {                         MvdL1[ x0 ][ y0 ][ 0 ] = 0                         MvdL1[ x0 ][ y0 ][ 1 ] = 0                     } else                         mvd_coding( x0, y0, 1 )                     mvp_11_flag[ x0 ][ y0 ]                 }             }         }     } } </pre> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 55.</p> <p><b>7.4.9.6 Prediction unit semantics</b></p> <p>...</p>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p><b>merge_idx[ x0 ][ y0 ]</b> specifies the merging candidate index of the merging candidate list where x0, y0 specify the location ( x0, y0 ) of the top-left luma sample of the considered prediction block relative to the top-left luma sample of the picture.</p> <p>When merge_idx[ x0 ][ y0 ] is not present, it is inferred to be equal to 0.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 107.</p> <p><b>8.5.3.2.2 Derivation process for luma motion vectors for merge mode</b></p> <p>...</p> <p>Outputs of this process are as follows, with X being 0 or 1:</p> <p>...</p> <ul style="list-style-type: none"> <li>- the reference indices refIdxLXA<sub>0</sub>, refIdxLXA<sub>1</sub>, refIdxLXB<sub>0</sub>, refIdxLXB<sub>1</sub> and refIdxLXB<sub>2</sub> of the neighbouring prediction units,</li> </ul> <p>...</p> <ul style="list-style-type: none"> <li>- the motion vectors mvLXA<sub>0</sub>, mvLXA<sub>1</sub>, mvLXB<sub>0</sub>, mvLXB<sub>1</sub> and mvLXB<sub>2</sub> of the neighbouring prediction units.</li> </ul> <p>...</p> <p>5. The merging candidate list, mergeCandList, is constructed as follows:</p> <pre>i = 0 if( availableFlagA<sub>1</sub> )     mergeCandList[ i++ ] = A<sub>1</sub> if( availableFlagB<sub>1</sub> )     mergeCandList[ i++ ] = B<sub>1</sub> if( availableFlagB<sub>0</sub> )     mergeCandList[ i++ ] = B<sub>0</sub> if( availableFlagA<sub>0</sub> )</pre>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<pre> mergeCandList[ i++ ] = A<sub>0</sub> if( availableFlagB<sub>2</sub> )     mergeCandList[ i++ ] = B<sub>2</sub> if( availableFlagCol )     mergeCandList[ i++ ] = Col  ... 9. The following assignments are made with N being the candidate at position merge_idx[ xOrigP ][ yOrigP ] in the merging candidate list mergeCandList ( N = mergeCandList[ merge_idx[ xOrigP ][ yOrigP ] ] ) and X being replaced by 0 or 1:     refIdxLX = refIdxLXN                                     (8-120)     predFlagLX = predFlagLXN                                 (8-121) 1. When use_integer_mv_flag is equal to 0 and the reference picture is not the current picture, the following applies:     mvLX[ 0 ] = mvLXN[ 0 ]                                     (8-122)     mvLX[ 1 ] = mvLXN[ 1 ]                                     (8-123) ... ITU-T Rec. H.265 (12/2016) High efficiency video coding at pp. 147-149.  <b>9    Parsing process</b> <b>9.1 General</b> Inputs to this process are bits ... Outputs of this process are syntax element values. This process is invoked when the descriptor of a syntax element in the syntax tables in clause 7.3 is equal to ue(v), se(v) (see clause 9.2), or ae(v) (see clause 9.3). </pre>

**EXHIBIT 9****UNITED STATES PATENT No. 10,536,714****CLAIM CHART FOR INFRINGEMENT OF CLAIM 9 BY ASUS ACCUSED PRODUCTS**

U.S. PATENT No. 10,536,714	ASUS ACCUSED PRODUCTS
	<p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 198.</p> <p><b>9.3 CABAC parsing process for slice segment data</b></p> <p><b>9.3.1 General</b></p> <p>This process is invoked when parsing syntax elements with descriptor ae(v) in clauses 7.3.8.1 through 7.3.8.11.</p> <p>Inputs to this process are a request for a value of a syntax element and values of prior parsed syntax elements.</p> <p>Output of this process is the value of the syntax element.</p> <p>ITU-T Rec. H.265 (12/2016) High efficiency video coding at p. 201.</p>